Simulation and Analysis of PFC Deadlock in Lossless Networks

Shani Budilovsky

February 2026

Abstract

Priority Flow Control (PFC) is a widely used mechanism in lossless Ethernet fabrics to support large-scale RDMA deployments. While PFC prevents packet loss by pausing upstream transmission when congestion occurs, it may lead to a severe failure mode known as deadlock, especially in the presence of cyclic buffer dependencies. In this work, we develop a Python-based discrete-time simulator that models PFC behavior in a simplified leaf-spine topology. We evaluate two scenarios: a baseline configuration with two RDMA-like flows that forms a cyclic dependency but remains stable, and a second configuration that adds an additional high-rate flow which triggers deadlock. The results show that deadlock occurs once all buffers in the dependency cycle cross the PFC threshold simultaneously, causing persistent pause propagation and preventing any forwarding. Our findings align with existing studies on PFC behavior and datacenter deadlocks, and highlight how traffic load and cyclic dependencies interact to create circular wait conditions.

1 Introduction

RDMA-based communication is increasingly deployed in modern datacenters due to its ability to deliver low-latency and high-throughput transport. Large-scale RDMA deployments over commodity Ethernet have been demonstrated in production environments, but require careful engineering of lossless fabrics to maintain performance and reliability [1].

Lossless Ethernet is often achieved through Priority Flow Control (PFC), which pauses upstream traffic when congestion is detected. While PFC reduces packet drops, it introduces new risks such as congestion spreading and deadlock. Deadlock is particularly problematic because it can freeze a portion of the network, preventing traffic from making progress even if offered load eventually decreases.

This report presents a simulator that demonstrates how PFC-induced deadlock can occur due to cyclic buffer dependencies. We evaluate two scenarios and analyze why one remains stable while the other leads to deadlock.

2 Background and Related Work

2.1 RDMA over Ethernet at Scale

Guo et al. [1] present a detailed study of deploying RDMA over commodity Ethernet at scale. Their work shows that RDMA can be deployed successfully in datacenters, but requires lossless behavior, careful queue management, and congestion control mechanisms. This motivates why PFC is commonly used, since packet drops are costly in RDMA environments.

2.2 Congestion Control for RDMA

Zhu et al. [3] propose DCQCN, a congestion control mechanism for large-scale RDMA deployments. Their key insight is that uncontrolled injection rates can lead to persistent congestion,

excessive PFC activation, and degraded throughput. DCQCN attempts to regulate sending rates using congestion feedback to prevent the network from reaching extreme congestion states.

This work is relevant to our simulator because our deadlock scenario is directly triggered by higher injection rates. The results highlight that congestion control is not only important for throughput, but also for avoiding pathological PFC behaviors.

2.3 Deadlocks in Datacenter Networks

Hu et al. [2] analyze deadlocks in datacenter networks and explain that deadlocks form when flow control mechanisms create cyclic dependencies between buffers. Their work emphasizes that deadlock is not simply a result of high congestion, but rather the combination of congestion with a structural dependency cycle that prevents draining.

The simulator in this project is directly inspired by the deadlock formation mechanism described in [2]. We implement a simplified cyclic dependency structure and evaluate how traffic load affects deadlock formation.

3 Methodology

3.1 The Setup

We implemented a discrete-time simulator in Python to model a leaf-spine topology. The network includes two spine switches and four leaf switches. Traffic is injected into the network through predefined flows. Each switch is modeled as having a finite buffer capacity.

To simplify the analysis and highlight the deadlock mechanism, the simulator models each switch as having a single shared buffer. In real hardware, ingress and egress queues are separated and host-facing egress ports may drain independently. However, the simplified model allows clear demonstration of cyclic dependency buildup.

3.2 PFC Model

At each time step, switches attempt to forward packets hop-by-hop. Forwarding from node u to node v is allowed only if the downstream buffer does not exceed the PFC threshold and is not full:

$$\text{occupancy}(v) < T_{PFC} \quad \text{and} \quad \text{occupancy}(v) < C$$

where T_{PFC} is the PFC threshold and C is the buffer capacity. If the downstream buffer occupancy reaches or exceeds T_{PFC}, forwarding is blocked, and the link is marked as paused.

3.3 Deadlock Detection

Deadlock is detected when all buffers in the cyclic dependency are at or above the PFC threshold, and no forwarding moves are possible within the cycle.
This definition follows the deadlock formation condition described in [2], where cyclic buffer dependencies combined with flow control can create a circular wait condition.

3.4 The Experiment

We evaluate two scenarios with identical cyclic dependency structure but different offered load. Both scenarios use:

- Buffer capacity: $C = 10$
- PFC threshold: $T_{PFC} = 7$

Scenario 1 includes two flows. Scenario 2 includes the same flows and adds an additional high-rate flow, increasing load into the cycle.

4 Results

4.1 Scenario 1: Cyclic Dependency Without Deadlock

Scenario 1 includes two flows which form a cyclic dependency.

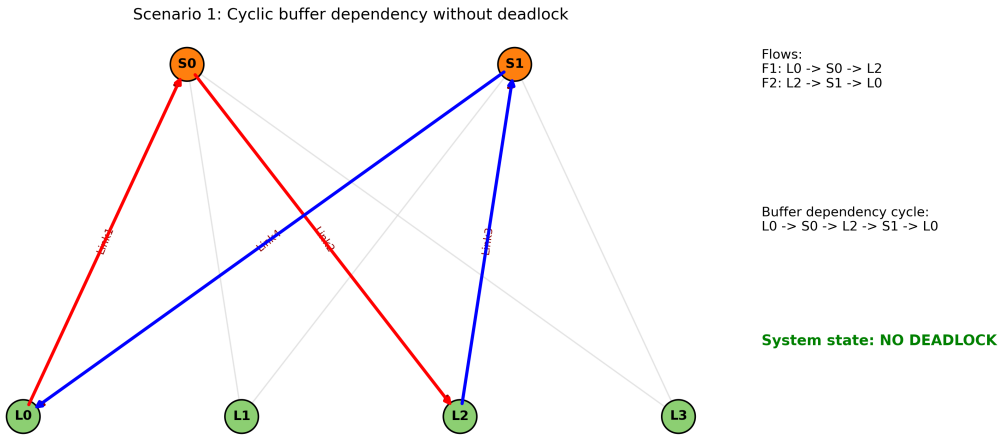

Figure 1: Scenario 1 topology with two flows forming a cyclic dependency without deadlock.

Figure 2 shows that buffers increase and approach the threshold but stabilize below full capacity and remain drainable..

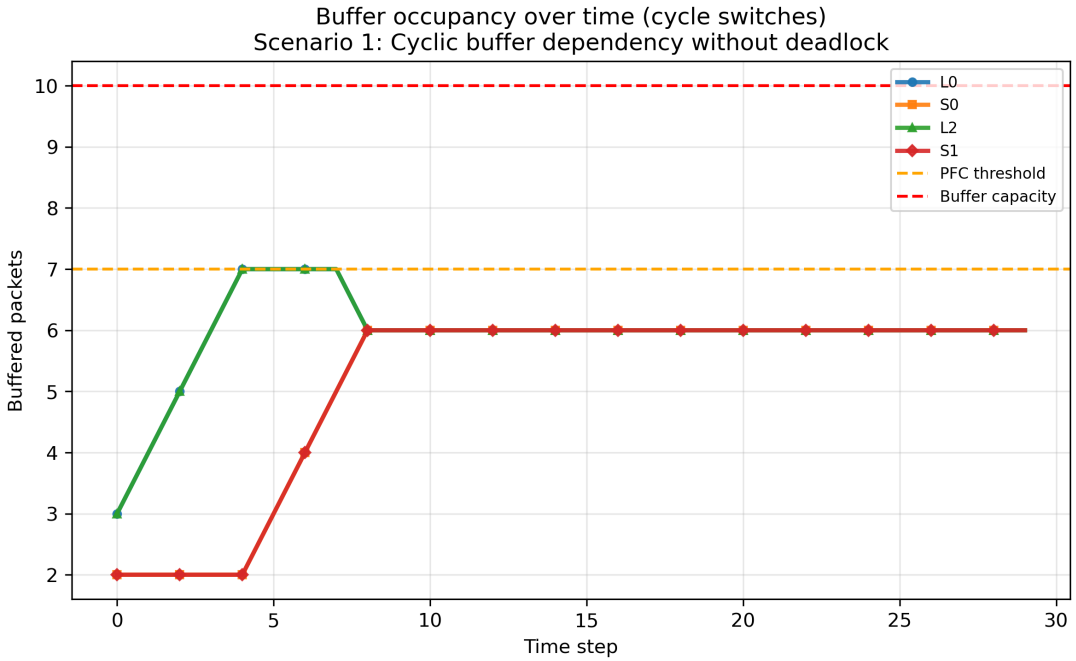

Figure 2: Scenario 1 buffer occupancy.

Figure 3 shows that pause events occur intermittently, but do not persist across all links.

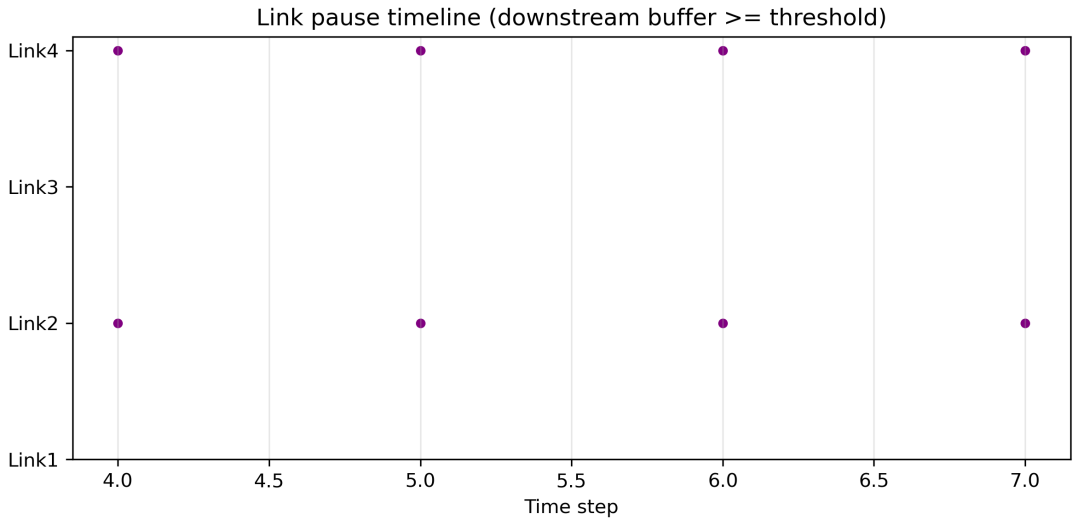

Figure 3: Scenario 1 link pause timeline.

Scenario 1 remains stable because at least one buffer in the cycle can still drain at each time step. The system does not reach a full cyclic wait condition, which is required for deadlock according to [2]. This demonstrates that cyclic dependency alone is not sufficient for deadlock; offered load plays a critical role.

4.2 Scenario 2: Additional Flow Causes Deadlock

Scenario 2 includes the same flows as Scenario 1 and adds a third flow. This additional flow injects more packets into the same cycle.

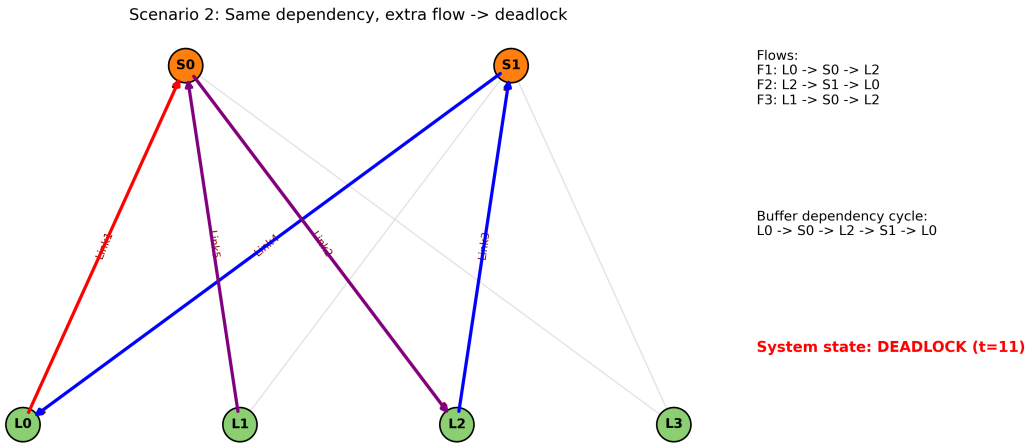

Figure 4: Scenario 2 topology.

Figure 5 shows that all buffers exceed the PFC threshold and approach capacity. So The cycle saturates due to higher injection load.

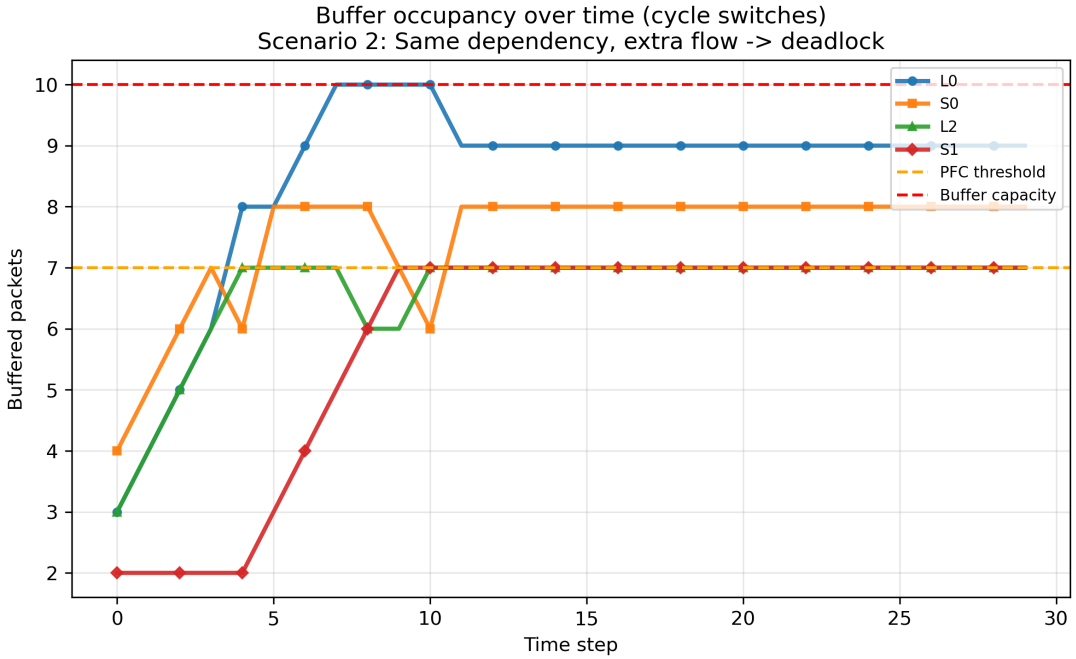

Figure 5: Scenario 2 buffer occupancy.

Figure 6 shows that all links eventually remain paused continuously.

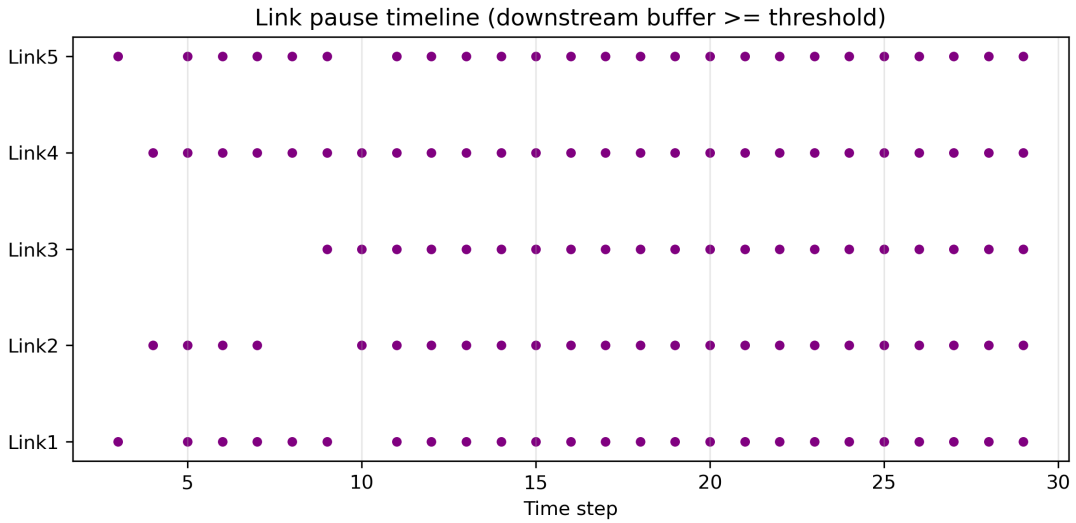

Figure 6: Scenario 2 link pause timeline. Persistent pauses indicate deadlock.

4.3 Deadlock Trigger

Deadlock is detected at $t = 11$. At this step, all buffers in the cyclic dependency cross the PFC threshold simultaneously. Since every downstream hop is congested, no forwarding is possible and the system freezes. This result matches the deadlock formation mechanism described in [2].

5 Discussion and Conclusion

This simulation demonstrates how PFC can induce deadlock when cyclic buffer dependencies exist and traffic load pushes all buffers in the cycle beyond the PFC threshold. Scenario 1 shows

that cyclic dependency does not necessarily imply deadlock; the offered load must be sufficiently high to saturate the entire cycle.

Scenario 2 shows that adding a single additional flow can trigger a tipping point, after which all links remain paused and forwarding becomes impossible. This emphasizes the importance of controlling injection rates.

The results connect directly to prior literature. The large-scale RDMA deployment work in [1] motivates the need for lossless fabrics, while DCQCN [3] demonstrates why congestion control is essential for preventing extreme buffer buildup. Our results highlight that without such control, PFC behavior can become pathological. The deadlock formation aligns with the analysis in [2], and the need for recovery mechanisms.

5.1 Limitations

The simulator models each switch as a single shared buffer and does not include separate ingress and egress draining behavior. In real switches, host-facing egress ports may continue draining even when internal buffers are paused. Therefore, the simulation exaggerates certain buildup behaviors. However, the fundamental circular wait mechanism remains valid and demonstrates the key deadlock condition.

5.2 Future Work

Possible improvements include:

- Modeling separate ingress and egress buffers
- Multi-priority PFC behavior
- Integrating congestion control similar to DCQCN [3]

References

[1] Chuanxiong Guo, Haitao Wu, Zhong Deng, Gaurav Soni, Jianxi Ye, Jitu Padhye, and Marina Lipshteyn. Rdma over commodity ethernet at scale. In *Proceedings of the 2016 ACM SIGCOMM Conference*, pages 202–215, 2016.

[2] Shuihai Hu, Yibo Zhu, Peng Cheng, Chuanxiong Guo, Kun Tan, Jitendra Padhye, and Kai Chen. Deadlocks in datacenter networks: Why do they form, and how to avoid them. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*, pages 92–98, 2016.

[3] Yibo Zhu, Haggai Eran, Daniel Firestone, Chuanxiong Guo, Marina Lipshteyn, Yehonatan Liron, Jitendra Padhye, Shachar Raindel, Mohamad Haj Yahia, and Ming Zhang. Congestion control for large-scale rdma deployments. *ACM SIGCOMM Computer Communication Review*, 45(4):523–536, 2015.